## 24.4 Packet-Switched Networks: Ethernet

### *24.4.1 Introduction*

In addition to traditional buses such as CAN or FlexRay, packet-switched Ethernet will be used in next-generation automotive communication architectures. Ethernet's superior bandwidth and flexibility make it ideal to address the high communication demands of, for example, Advanced Driver Assistance Systems (ADASs), infotainment systems, and ECU flashing. As a switched network, Ethernet provides a scalable, high-speed, and cost-effective communication platform, which allows arbitrary topologies.

Ethernet evolved from a shared bus communication medium with CSMA/CD-based link access scheme to a switched network. Frame collisions in CSMA/CD were resolved by a binary exponential backoff algorithm which picked a random delay until a retransmission could be started after a collision. This deemed CSMA/CD unsuitable for real-time systems with tight latency or jitter requirements. Switched Ethernet made CSMA/CD obsolete. In switched Ethernet, contention is moved into the switches, where a scheduler has full control over each output port. This enables the implementation of elaborate link schedulers, which allow the derivation of real-time guarantees. Today, Ethernet installations (including the automotive domain) are almost always switched. Hence, in the following, we will refer to switched Ethernet as standard Ethernet.

In the automotive context, Ethernet is anticipated to serve as an in-vehicle communication backbone, where it must be able to transport traffic streams of mixed-criticality. This requires Quality of Service (QoS) mechanisms, in order to provide deterministic timing guarantees for critical traffic. Standard Ethernet (IEEE 802.1Q) introduced eight traffic classes. These classes can be used to prioritize traffic, which is typically implemented by a Static-Priority Non-Preemptive (SPNP) scheduler at each output port in each switch and end-point. This limited number of classes requires that multiple traffic streams share a class, making streams of equal priority indistinguishable to the scheduler. Traffic within a shared class is usually scheduled in First In First Out (FIFO) order.

Compared to CAN or FlexRay, Ethernet exhibits complex timing behavior, as each switch output port is a point of arbitration, which adds delay to the overall end-to-end latency. While mature formal performance analysis techniques have been established for CAN and FlexRay, such techniques are even more required for Ethernet before it can be used in timing- and safety-critical systems. This will become even more important in the context of highly automated and autonomous driving. In this section, we use Compositional Performance Analysis (CPA) (see Chapter 23 and [29]) to derive worst-case performance bounds for Ethernet.

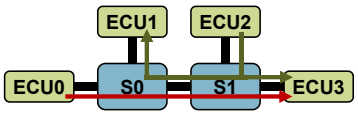


Figure 24.6: Ethernet model

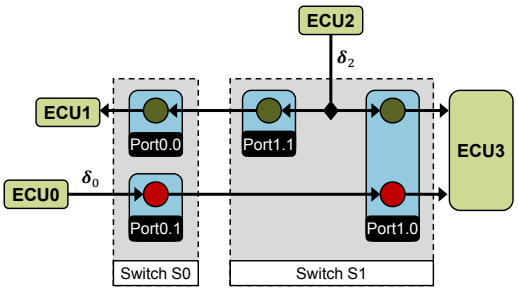


Figure 24.7: CPA system model for the Ethernet model from Figure 24.6

### *24.4.2 Modeling Ethernet Networks for Performance Analysis*

Before timing guarantees can be derived for an Ethernet network, the components of this network must be mapped to the CPA system model (cf. Section 23.2). This mapping process is explained in detail in [19]. Here, a brief summary covering the essential steps, using Figures 24.6 and 24.7 as illustration is presented.

Figure 24.6 shows an Ethernet model comprising two switches and four ECUs. A sequence of related Ethernet frames between a source and one (or more) destination(s) is called an Ethernet traffic stream. There are two traffic streams in the network: a unicast stream from ECU0 to ECU3 and a multicast stream from ECU2 to ECU1 and ECU3.

In order to map the Ethernet model to the CPA system model, resources, tasks, and event models must be identified. Resources model points of contention. In Ethernet, contention between individual frames happens at the switches. Inside a switch there are several delay sources. At the input port, there is *input queuing delay*, the switch fabric adds *forwarding delay*, and at the output port, there is *output queuing delay*. Contemporary switches are fast enough that input queuing delay and forwarding delay only have a negligible impact on the overall timing guarantees. Hence, these delays can be ignored or approximated by constant terms. The output queuing delay considers the time it takes to transmit a given frame, including the interference from other frames. Consequently, switch output ports are modeled by CPA resources. The scheduling policy of these resources is determined by the switch port's scheduling mechanism. Additionally, here is *transmission delay* on the link between switches. This delay corresponds to the propagation delay of electric signals on the link's wire and can also be modeled by a constant term.

The transmission of a frame via an output port of a switch is modeled in CPA as the execution of a task on the port's resource. An Ethernet traffic stream is modeled as a chain of dependent frames (tasks) according to its path through the network (see Figure 24.7). This chain may fork to model multicast or broadcast trees. On each resource, a task consumes service according to its execution time bounds, which are derived from the best-case and worst-case transmission times of its corresponding Ethernet frame. The transmission time of a frame is defined to be the time it takes the frame to be transmitted without any interference from other frames. For a frame of traffic stream $i$ with maximum/minimum payload $p_i^{-/+}$, the best-case and worst-case transmission times $C_i^-$ and $C_i^+$ can be computed to:

$$C_i^{+/-} = \frac{42\,\text{bytes} + \max\left\{42\,\text{bytes}, p_i^{+/-}\right\}}{r_{TX}} \tag{24.45}$$

where $r_{TX}$ is the transmission rate of the port that transmits the frame. The constant terms correspond to the protocol overhead. The first 42 bytes account for preamble (7 bytes), start of frame delimiter (1 byte), destination and source MAC address (both 6 bytes), IEEE 802.1Q tag (4 bytes), EtherType (2 bytes), frame check sequence (4 bytes), and inter-frame gap (12 bytes). The second 42 bytes account for the fact that there is a minimum Ethernet frame size of 84 bytes and that the payload must be padded if necessary.

Frame arrivals (and emissions) are modeled by event models. These models come from either external sources or from dependent frames.

Figure 24.7 shows the corresponding CPA model of the Ethernet model from Figure 24.6. As can be seen, the output ports of both switches are modeled as resources (light blue boxes with rounded corners). Both traffic streams are modeled by a chain of tasks (red and green circles) reflecting their paths through the network. Notice that the green path originating at ECU2 splits into a multicast tree. ECU0 and ECU2 inject frames into the network according to the event models $\delta_0$ and $\delta_2$ (respectively). This model can then be analyzed with CPA's iterative approach.

### *24.4.3 Analysis of Standard Ethernet (IEEE 802.1Q)*

In order to derive upper bounds on the worst-case performance of Ethernet networks, an analysis which captures all delay effects on the CPA resources that model the switch output ports must be developed. This analysis will then be used in the local analysis step of the CPA loop to derive worst-case frame transmission latencies on each output port.

**Definition 24.1.** A frame's transmission latency is the time interval, which starts when the frame has been received at an input port and ends when it has been transmitted entirely from an output port. The transmission latency includes all timing effects from interfering traffic streams.

In the context of the model transformation from Section 24.4.2, the transmission latency of a frame corresponds to the response time of a task.

When deriving formal performance guarantees, the worst-case transmission latency of the frames of a given traffic stream $i$ (among all for stream $i$'s possible frames transmission latencies) is of particular interest. For non-preemptive scheduling (such as standard Ethernet), it has been shown that, in order to find this worst-case transmission latency, the transmission latencies of all frames of stream $i$ in its longest scheduling horizon must be evaluated (cf. [12]). The scheduling horizon of a traffic stream $i$ is the time a switch port is busy processing frames of stream $i$, including interference from frames of other traffic streams (cf. Section 23.2.2.1 and [17]). Particularly, the worst-case transmission latency of the $q$-th frame of traffic stream $i$, can be derived from its worst-case multiple activation queuing delay $Q_i(q, a_i^q)$ (cf. Eq. (23.9) in Section 23.2.2.1).

**Definition 24.2.** Assuming that the $q$-th frame of a traffic stream $i$ arrives at time $a_i^q$ at a switch output port, its worst-case multiple activation queuing delay $Q_i(q, a_i^q)$ is the time interval, which starts with the arrival of the first frame of stream $i$ that initiates the scheduling horizon and ends when the $q$-th frame can be transmitted (i.e. it does not include the transmission of the $q$-th frame).

Note that, in contrast to the multiple activation queuing delay $Q_i(q)$ introduced in Section 23.2.2.1, the queuing delay in the Ethernet context additionally depends on the arrival time $a_i^q$ of the $q$-th frame. This is due to the FIFO scheduling of frames with equal priority and will be explained later in this section. The arrival time $a_i^q$ of the $q$-th frame of stream $i$ is measured relative to the beginning of the scheduling horizon.

As stated in Section 24.4.2, it is assumed that the queuing delay of a given frame at a switch output port accounts for all delays induced by interfering traffic streams. The amount of interference from other traffic streams depends on the output port's scheduling policy. In standard Ethernet, traffic streams are categorized into (up to) eight traffic classes, which correspond to priority levels. Inside each output port there is a set of FIFO queues, one for each traffic class. These FIFO queues are served by an SPNP scheduler. Consequently, to calculate the worst-case queuing delay $Q_i(q, a_i^q)$ in standard Ethernet, all blocking effects, which can occur in this combination of FIFO and SPNP scheduling, must be considered.

**Lower-priority blocking**: In non-preemptive scheduling, a frame which started transmitting is guaranteed to finish without interruption. Hence, a frame of traffic stream $i$ can experience blocking from at most one lower-priority frame, if this lower-priority frame started transmitting just before the arrival of the first frame of traffic stream $i$ [21].

$$I_i^{LPB} = \max_{j \in lp(i)} \left\{ C_j^+ \right\} \tag{24.46}$$

where $lp(i)$ is a function yielding the set all traffic streams whose priority is lower than that of stream $i$.

**Higher-priority blocking**: In any time interval of length $\Delta t$, a frame of traffic stream $i$ can experience blocking from all frames of higher-priority streams, which

arrive during $\Delta t$, i.e. before the frame of stream $i$ can be transmitted [21].

$$I_i^{HPB}(\Delta t) = \sum_{j \in hp(i)} \eta_j^+(\Delta t + \varepsilon) C_j^+ \tag{24.47}$$

where $hp(i)$ is a function yielding the set all traffic streams whose priority is higher than that of stream $i$. Recall from Section 23.2.1.2 that $\eta^+(\Delta t)$ yields a upper bound on the number of events, i.e. frame arrivals, in any half open time interval of length $\Delta t$. As the multiple activation queuing delay $Q_i(q, a_i^q)$ covers the time *until* the $q$-th frame can be transmitted, higher-priority frames arriving exactly at the end for $\Delta t$ can also interfere with the $q$-th frame. We model this by adding an infinitesimal small time $\varepsilon$ to $\Delta t$ to cover the *closed* time interval $[t, t + \Delta t]$. In practice, $\varepsilon$ corresponds to a bit time.

**Same-priority blocking**: As frames of identical priority are processed in FIFO order, frames of traffic stream $i$ can experience blocking from frames of other traffic streams with the same priority as stream $i$. Hence, if the $q$-th frame of traffic stream $i$ arrives at time $a_i^q$, it must wait for all frames from other streams with identical priority, which arrived before or at $a_i^q$, as well as wait for its own $q-1$ predecessor frames to finish [21].

$$I_i^{SPB}(q, a_i^q) = (q-1)C_i^+ + \sum_{j \in sp(i)} \eta_j^+(a_i^q + \varepsilon) C_j^+ \tag{24.48}$$

Here, $sp(i)$ is a function yielding the set all traffic streams whose priority is equal to that of stream $i$ (excluding stream $i$). In the worst-case, any same-priority frames arriving concurrently at exactly $a_i^q$, are assumed to interfere with the $q$-frame of stream $i$. Again, an infinitesimal small time $\varepsilon$ is added to $\Delta t$ to cover this case.

In [21] it is shown that FIFO scheduling requires a candidate search in order to determine the worst-case blocking. The reason for this candidate search is that if frame $q$ arrives early (within its jitter bounds), it might experience additional blocking from some of its own $q-1$ queued predecessors. However, if it arrives late (within its jitter bounds), it might experience additional blocking from previously queued frames of interfering same-priority streams. The set of arrival candidates $A_i^q$ can be reduced to points in time where the candidates $a_i^q$ coincide with the earliest arrivals of interfering frames from same-priority traffic streams [21]. Consequently, all candidates for the arrival of the $q$-th frame of stream $i$ can be found by investigating the arrivals of interfering frames between the earliest arrival $\delta_i^-(q)$ of the $q$-th frame and its $q$-activation scheduling horizon $S_i(q)$, which is the time a switch port is busy processing $q$ frames of stream $i$, including interfering frames from other traffic streams (cf. Eq. (23.4 in Section 23.2.2)):

$$A_i^q = \bigcup_{j \in sp(i)} \left\{ \delta_j^-(n) | \delta_i^-(q) \leq \delta_j^-(n) < S_i(q) \right\}_{n \geq 1} \tag{24.49}$$

where, in the context of standard Ethernet, $S_i(q)$ can be computed as follows:

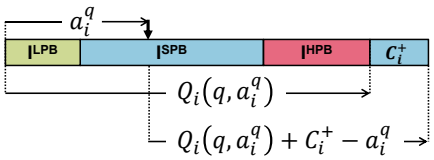


Figure 24.8: Example queuing delay and transmission latency computation (cf. [75])

$$S_i(q) = \max_{j \in lp(i)} \left\{ C_j^+ \right\} + qC_i^+ + \sum_{j \in sp(i) \cup hp(i)} \eta_j^+ \left( S_i(q) \right) C_j^+ \tag{24.50}$$

Note that the computation of the $q$-activation scheduling horizon does not require a candidate search for the same-priority interference, as it is only concerned about the time when the port is busy. As $S_i(q)$ occurs on both sides, Eq. (24.50) cannot be solved directly. However, it represents an integer fixed-point problem, which can be solved by iteration, as all terms are monotonically increasing (cf. [29]). A valid starting point is e.g. $S_i(q) = \max_{j \in lp(i)} \{C_j^+\} + qC_i^+$.

In order to compute the worst-case queuing delay $Q_i(q, a_i^q)$ of the $q$-th frame arrival of traffic stream $i$, which arrived at time $a_i^q$, all presented blocking effects must be considered:

$$Q_i(q, a_i^q) = I_i^{LPB} + I_i^{SPB}(q, a_i^q) + I_i^{HPB}\left( Q_i(q, a_i^q) \right) \tag{24.51}$$

Again, $Q_i(q, a_i^q)$ occurs on both sides and Eq. (24.51) cannot be solved directly. Like the integer fixed-point problem in Eq. (24.50), it can be solved by iteration with e.g. $Q_i(q, a_i^q) = (q-1)C_i^+$ as a starting point.

Now, the largest transmission latency $R_i(q)$ for the $q$-th frame arrival of traffic stream $i$ can be computed by adding the transmission time $C_i^+$ of this $q$-th frame to its worst-case queuing delay and accounting for the fact that the frame arrived at time $a_i^q$ (see e.g. [3]). This is illustrated in Figure 24.8.

$$R_i(q) = \max_{a_i^q \in A_i^q} \left\{ Q_i(q, a_i^q) + C_i^+ - a_i^q \right\} \tag{24.52}$$

By taking the maximum over all $R_i(q)$, the worst-case frame transmission latency for a frame of stream $i$ can be computed:

$$R_i^+ = \max_{1 \le q \le q_i^+} \{R_i(q)\} \tag{24.53}$$

As mentioned before, in order to derive the worst-case frame transmission latency of stream $i$, all frame arrivals of stream $i$ in its longest scheduling horizon must be evaluated. Let $q_i^+$ be the maximum number of these frame arrivals. It can be derived by computing the maximum number of frames, which arrive during the scheduling horizon of their respective predecessors (cf. [18]).

$$q_i^+ = \max_{q \geq 1} \left\{ q | \delta_i^-(q) \leq S_i(q-1) \right\} \tag{24.54}$$

Now, as established in Sections 23.2.2.1 and 23.2.2.2, the worst-case bounds on the maximum path latency and the maximum frame backlog can be derived from the maximum frame transmission latencies and maximum $q$-activation processing times.

#### 24.4.3.1 End-to-End Latency Bounds

From the individual worst-case transmission latencies of the frames along the path of a traffic stream through the network, the worst-case end-to-end latency of the stream can be derived. Let $Path(i)$ be the path of stream $i$ through the network. Now, the time it takes to transmit $q$ frames of stream $i$, i.e. its worst-case $q$ activation end-to-end latency, can be bounded by (cf. [20]):

$$L_i^+(q) = \delta_i^-(q) + \sum_{j \in Path(i)} R_j^+ \tag{24.55}$$

Here, the frames of stream $i$ are injected into the network at their maximum rate (i.e. with minimum inter-arrival times $\delta_i^-(q)$) to induce maximum load on the system's resources. Along any given path, frames of a traffic stream are processed in order, i.e. they cannot overtake each other. Eq. (24.55) assumes that the last of the $q$ frames experiences the worst-case transmission latency on all its ports. Due to in-order processing, all $q-1$ previously sent frames must have arrived by then.

Obviously, for $q = 1$, Eq. (24.55) yields the worst-case end-to-end latency of a single frame (recall that $\delta_i^-(1) = 0$). Larger $q$ are convenient in cases where, for example, a large IP packet is distributed over multiple Ethernet frames.

#### 24.4.3.2 Buffer Size Bounds

Apart from timing guarantees, buffer size requirements are also important, as actual systems (e.g. switches) only have limited memory resources (buffer space). Insufficient buffer space can lead to frame drop, which is highly undesirable for (time) critical traffic.

The maximum activation backlog of a traffic stream $i$ is an upper bound on the number of frames from $i$ that can be queued at a resource at any given time. It can be derived by computing, for each $q$-th frame, the maximum number of frames, which arrived until the $q$-th frame has been transmitted, and subtracting from this number the $q-1$ frames that must have been transmitted prior to the $q$-th one (cf. [21]).

$$b_i^+ = \max_{1 \leq q \leq q_i^+} \left\{ \eta_i^+ \left( B_i^+(q) \right) - q + 1 \right\} \tag{24.56}$$

where $B_i^+(q)$ is the multiple activation processing time. Given $q$ consecutive frames of a traffic stream $i$, the multiple activation processing time is the longest time interval between the arrival of the first frame and end of the transmission of the $q$-th frame (cf. Eq. (23.8) in Section 23.2.2.1).

$$B_i^+(q) = Q_i(q) + C_i^+ \tag{24.57}$$

In the Ethernet context, it can be bounded by the multiple activation queuing delay under the assumption that all event arrive as soon as possible (i.e. we do not need to consider different event arrivals as in Eq. (24.51)) by adding the frames worst-case transmission time $C_i^+$ (cf. Eq. (23.9)):

$$Q_i(q) = I_i^{LPB} + I_i^{SPB}(q, Q_i(q)) + I_i^{HPB}(Q_i(q)) \tag{24.58}$$

From the maximum activation backlogs, the maximum buffer size requirements can be derived. Typically, the memory in Ethernet switches can only be allocated block-wise, e.g. in blocks of 128 bytes or 256 bytes. This must be taken into account when deriving the maximum buffer size requirements. Assuming that a switch only allows the allocation of memory blocks of size $m$ and that only the destination and source MAC addresses, the IEEE 802.1Q tag, the EtherType, the maximum payload $p^+$, and the frame check sequence of an Ethernet frame must be stored in switch memory, the buffer size requirement (in bytes) for a traffic stream $i$ can be bounded by (cf. Section 24.4.2):

$$\hat{b}_i^+ = b_i^+ \left\lceil \frac{22\,\text{bytes} + \max\{42\,\text{bytes}, p^+\}}{m} \right\rceil m \tag{24.59}$$

The buffer size requirement per port can be computed by summing the buffer size requirements of all streams passing this port, and the buffer size requirement of a switch can be computed by summing up the requirements of each of its ports.

### *24.4.4 Analysis Extensions*

#### 24.4.4.1 Other Ethernet Schedulers

As Ethernet strives to cover a wide range of application domains, it supports many different schedulers and shapers to forward frames, each of which has a different impact on the queuing delay at a switch's output port. For the most prominent ones, CPA-based analyses are available.

Ethernet AVB [31] introduced standardized traffic shaping in the form of credit-based shaping on top of standard Ethernet. The motivation is to shape higher-priority traffic streams to bound their interference on lower-priority ones, e.g. to prevent starvation. However, as any form of traffic shaping introduces additional delays, a careful timing analysis is required to evaluate Ethernet AVB's applicability for

real-time applications. Formal analyses for Ethernet AVB in the context of the CPA framework are presented in [21] and [3].

Ethernet TSN defines a set of Ethernet standards, which were designed with real-time requirements in mind. Some of these standards specify new link arbitration mechanisms. Namely, IEEE 802.1Qbv [35] introduces time-triggered frame forwarding to Ethernet, i.e. frames of time-triggered traffic classes are scheduled at predefined points in time such that they do not experience interference from other traffic classes. IEEE 802.1Qbv relies on so called guard bands to block non-time-triggered traffic early enough to prevent interference with time-triggered traffic. In IEEE 802.1Qch [32], cyclic frame forwarding is defined, i.e. frame forwarding is based on alternating time intervals and frames received in one interval, will be sent in the next interval etc. A new credit-based shaper, which aims to improve the forwarding of bursts, is discussed in [25]. Formal analyses for these shapers are presented in [75] and [74].

Although not explicitly standardized by the IEEE, weighted round robin scheduling can be implemented as an IEEE 802.1Q enhanced transmission selection algorithm. A CPA-compatible formal analysis for weighted round robin scheduling in the Ethernet context has been presented in [71].

In order to improve the timing of critical traffic, frame preemption has been introduced to Ethernet via the IEEE 802.3br [33] and IEEE 802.1Qbu [34] standards. A CPA-compatible formal analysis for frame preemption has been presented in [73].

#### 24.4.4.2 Analysis Improvements

This section covered the fundamental approach to derive timing guarantees for Ethernet networks in CPA. The presented baseline analysis has been improved and extended in many directions.

Different analysis optimizations to exploit various kinds of correlations between Ethernet traffic streams have been proposed in [3] and [70]. [3] exploits the fact that both Ethernet links and Ethernet AVB's traffic shapers limit the amount of workload, which can pass them in a given time interval. This property can be used to limit the interference during the computation of the worst-case frame transmission latencies. In [70], the authors show how FIFO scheduling can be exploited to reduce the interference a frame can experience from its same-priority predecessors.

#### 24.4.4.3 Higher-Layer Protocols

Ethernet only defines frame forwarding on layer 2 of the ISO/OSI model. Higher-layer protocols often have additional timing implications. In [2] and [72] analyses to determine a bound on the worst-case timing impact of automatic repeat requests (ARQ) and software-defined networking (SDN) [42] are presented.

Due to the compositional nature of CPA, the Ethernet analysis can be easily combined with other analyses from the CPA framework to derive system-wide perfor-

mance guarantees. In [76], this has been done to compute end-to-end latency bounds for CAN-over-Ethernet traffic, where Ethernet ports are modeled as described in this section, but CAN buses and gateway processors are modeled according to their respective scheduling policies.

**Acknowledgements** The contribution *Packet-Switched Networks: Ethernet* has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 644080.

## References

1. Andersson, B., Tovar, E.: The utilization bound of non-preemptive rate-monotonic scheduling in controller area networks is 25%. In: 2009 IEEE International Symposium on Industrial Embedded Systems, pp. 11–18 (2009)
2. Axer, P., Thiele, D., Ernst, R.: Formal Timing Analysis of Automatic Repeat Request for Switched Real-Time Networks. In: In Proc. of SIES. Pisa, Italy (2014)
3. Axer, P., Thiele, D., Ernst, R., Diemer, J.: Exploiting Shaper Context to Improve Performance Bounds of Ethernet AVB Networks. In: Proceedings of DAC. San Francisco (2014)
4. Baruah, S., Chen, D., Gorinsky, S., Mok, A.: Generalized multiframe tasks. Real-Time Systems **17**(1), 5–22 (1999)
5. Broster, I., Burns, A., Rodriguez-Navas, G.: Probabilistic analysis of can with faults. In: 23rd IEEE Real-Time Systems Symposium, pp. 269–278 (2002)
6. von der Bruggen, G., Chen, J.J., Huang, W.H.: Schedulability and optimization analysis for non-preemptive static priority scheduling based on task utilization and blocking factors. In: Real-Time Systems (ECRTS), 2015 27th Euromicro Conference on, pp. 90–101. IEEE (2015)
7. Casparsson, L., Rajnak, A., Tindell, K., Malmberg, P.: Volcano revolution in on-board communications. Tech. rep., Volvo (1998)
8. Chen, Y., Kurachi, R., Takada, H., Zeng, G.: Schedulability comparison for can message with offset: Priority queue versus fifo queue. In: 19th International Conference on Real-Time and Network Systems, pp. 181–192 (2011)
9. Darbandi, A., Kim, M.K.: Schedule optimization of static messages with precedence relations in flexray. In: Sixth International Conference on Ubiquitous and Future Networks, pp. 495–500 (2014)
10. Darbandi, A., Kwon, S., , Kim, M.K.: Scheduling of time triggered messages in static segment of flexray. International Journal of Software Engineering and Its Applications **8**(6), 195–208 (2014)
11. Davis, R., Navet, N.: Controller area network (CAN) schedulability analysis for messages with arbitrary deadlines in FIFO and work-conserving queues. In: 9th IEEE International Workshop on Factory Communication Systems, pp. 33–42 (2012)
12. Davis, R.I., Burns, A., Bril, R.J., Lukkien, J.J.: Controller Area Network (CAN) schedulability analysis: Refuted, revisited and revised. Real-Time Systems **35**(3), 239–272 (2007)
13. Davis, R.I., Kollmann, S., Pollex, V., Slomka, F.: Schedulability Analysis for Controller Area Network (CAN) with FIFO Queues Priority Queues and Gateways. Real-Time Systems **49**(1), 73–116 (2013)
14. Di Natale, M., Zeng, H.: System identification and extraction of timing properties from controller area network (CAN) message traces. In: IEEE Conference on Emerging Technologies and Factory Automation, pp. 1–8 (2010)
15. Di Natale, M., Zeng, H.: Practical issues with the timing analysis of the controller area network. In: 18th IEEE Conference on Emerging Technologies Factory Automation, pp. 1–8 (2013)

16. Di Natale, M., Zeng, H., Giusto, P., Ghosal, A.: Understanding and using the controller area network communication protocol: theory and practice. Springer Science & Business Media (2012)
17. Diemer, J.: Predictable network-on-chip for general-purpose processors – formal worst-case guarantees for on-chip interconnects. Ph.D. thesis, Technische Universität Braunschweig, Braunschweig, Germany. URL N/A. To appear
18. Diemer, J., Axer, P., Ernst, R.: Compositional Performance Analysis in Python with pyCPA. In: Int. Workshop on Analysis Tools and Methodologies for Embedded and Real-time Systems (2012)
19. Diemer, J., Rox, J., Ernst, R.: Modeling of Ethernet AVB Networks for Worst-Case Timing Analysis. In: MATHMOD - Vienna International Conference on Mathematical Modelling. Vienna, Austria (2012)
20. Diemer, J., Rox, J., Negrean, M., Stein, S., Ernst, R.: Real-time communication analysis for networks with two-stage arbitration. In: Proceedings of the ninth ACM International Conference on Embedded Software (EMSOFT), EMSOFT 2011, pp. 243–252. ACM, Taipei, Taiwan (2011)
21. Diemer, J., Thiele, D., Ernst, R.: Formal Worst-Case Timing Analysis of Ethernet Topologies with Strict-Priority and AVB Switching. In: IEEE International Symposium on Industrial Embedded Systems (2012). Invited Paper
22. Ding, S.: Scheduling approach for static segment using hybrid genetic algorithm in flexray systems. In: 10th IEEE International Conference on Computer and Information Technology, pp. 2355–2360 (2010)
23. Ding, S., Murakami, N., Tomiyama, H., Takada, H.: A ga-based scheduling method for flexray systems. In: 5th ACM International Conference on Embedded Software, pp. 110–113 (2005)
24. Ghosal, A., Zeng, H., Di Natale, M., Ben-Haim, Y.: Computing robustness of flexray schedules to uncertainties in design parameters. In: Proceedings of the Conference on Design, Automation and Test in Europe, pp. 550–555 (2010)
25. Götz, F.J.: Alternative Shaper for Scheduled Traffic in Time Sensitive Networks. In: IEEE 802.1 TSN TG Meeting. Vancouver, BC, Canada (2013)
26. Grenier, M., Havet, L., Navet, N.: Configuring the communication on flexray-the case of the static segment. In: 4th European Congress on Embedded Real Time Software (2008)
27. Han, G., Di Natale, M., Zeng, H., Liu, X., Dou, W.: Optimizing the implementation of real-time simulink models onto distributed automotive architectures. Journal of Systems Architecture **59**(10), 1115–1127 (2013)
28. Han, G., Zeng, H., Li, Y., Dou, W.: Safe: Security-aware flexray scheduling engine. In: Design, Automation and Test in Europe Conference and Exhibition (2014)
29. Henia, R., Hamann, A., Jersak, M., Racu, R., Richter, K., Ernst, R.: System Level Performance Analysis - the SymTA/S Approach. In: IEE Proceedings Computers and Digital Techniques (2005)
30. Hu, M., Luo, J., Wang, Y., Lukasiewycz, M., Zeng, Z.: Holistic scheduling of real-time applications in time-triggered in-vehicle networks. IEEE Transactions on Industrial Informatics **10**(3), 1817–1828 (2014)
31. IEEE Audio Video Bridging Task Group: 802.1Qav - Forwarding and Queuing Enhancements for Time-Sensitive Streams. http://www.ieee802.org/1/pages/802.1av.html (2010)
32. IEEE Audio Video Bridging Task Group: 802.1Qch - Cyclic Queuing and Forwarding. http://www.ieee802.org/1/pages/802.1ch.html (2016)
33. IEEE P802.3br Interspersing Express Traffic Task Force: P802.3br - Standard for Ethernet Amendment Specification and Management Parameters for Interspersing Express Traffic. https://standards.ieee.org/develop/project/802.3br.html
34. IEEE Time-Sensitive Networking Task Group: 802.1Qbu - Frame Preemption. http://www.ieee802.org/1/pages/802.1bu.html
35. IEEE Time-Sensitive Networking Task Group: P802.1Qbv (Draft 3.0) - Enhancements for Scheduled Traffic. http://www.ieee802.org/1/pages/802.1bv.html (2015)

36. International Standards Organisation (ISO): ISO 11898-1. Road vehicles - interchange of digital information - controller area network (CAN) for high-speed communication. ISO Standard-11898 (1993)
37. International Standards Organisation (ISO): Road vehicles – FlexRay communications system – Part 1: General information and use case definition. ISO Standard-17458 (2013)
38. Jansen, K., Solis-Oba, R.: An asymptotic fully polynomial time approximation scheme for bin covering. Theoretical Computer Science **306**(1), 543–551 (2003)
39. Kang, M., Park, K., Jeong, M.K.: Frame packing for minimizing the bandwidth consumption of the flexray static segment. IEEE Transactions on Industrial Electronics **60**(9), 4001–4008 (2013)
40. Khan, D., Bril, R., Navet, N.: Integrating hardware limitations in can schedulability analysis. In: 8th IEEE International Workshop on Factory Communication Systems, pp. 207–210 (2010)
41. Khan, D., Davis, R., Navet, N.: Schedulability analysis of can with non-abortable transmission requests. In: 16th IEEE Conference on Emerging Technologies Factory Automation, pp. 1–8 (2011)
42. Kreutz, D., Ramos, F., Esteves Verissimo, P., Esteve Rothenberg, C., Azodolmolky, S., Uhlig, S.: Software-defined networking: A comprehensive survey. Proceedings of the IEEE **103**(1), 14–76 (2015)
43. Li, W., Di Natale, M., Zheng, W., Giusto, P., Sangiovanni-Vincentelli, A., Seshia, S.: Optimizations of an application-level protocol for enhanced dependability in flexray. In: Design, Automation Test in Europe Conference Exhibition, 2009. DATE '09., pp. 1076–1081 (2009)
44. Lincoln, B., Cervin, A.: Jitterbug: a tool for analysis of real-time control performance. In: Proceedings of the 41st IEEE Conference on Decision and Control, vol. 2, pp. 1319–1324 (2002)
45. Liu, M., Behnam, M., Nolte, T.: An evt-based worst-case response time analysis of complex real-time systems. In: 8th IEEE International Symposium on Industrial Embedded Systems, pp. 249–258 (2013)
46. Liu, M., Behnam, M., Nolte, T.: Schedulability analysis of multi-frame messages over controller area networks with mixed-queues. In: 18th IEEE Conference on Emerging Technologies Factory Automation, pp. 1–6 (2013)
47. Liu, M., Behnam, M., Nolte, T.: Schedulability analysis of gmf-modeled messages over controller area networks with mixed-queues. In: 10th IEEE Workshop on Factory Communication Systems, pp. 1–10 (2014)
48. Lukasiewycz, M., Glaß, M., Teich, J., Milbredt, P.: Flexray schedule optimization of the static segment. In: 7th IEEE/ACM International Conference on Hardware/Software Codesign and System Synthesis, pp. 363–372 (2009)
49. Lukasiewycz, M., Schneider, R., Goswami, D., Chakraborty, S.: Modular scheduling of distributed heterogeneous time-triggered automotive systems. In: 17th Asia and South Pacific Design Automation Conference, pp. 665–670 (2012)
50. Mok, A., Chen, D.: A multiframe model for real-time tasks. In: 17th IEEE Real-Time Systems Symposium, pp. 22–29 (1996)
51. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Extending schedulability analysis of controller area network (can) for mixed (periodic/sporadic) messages. In: 16th IEEE Conference on Emerging Technologies Factory Automation, pp. 1–10 (2011)
52. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Extending response-time analysis of mixed messages in can with controllers implementing non-abortable transmit buffers. In: 17th IEEE Conference on Emerging Technologies Factory Automation, pp. 1–4 (2012)
53. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Response time analysis for mixed messages in can supporting transmission abort requests. In: 7th IEEE International Symposium on Industrial Embedded Systems, pp. 291–294 (2012)
54. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Response-time analysis of mixed messages in controller area network with priority- and fifo-queued nodes. In: 9th IEEE International Workshop on Factory Communication Systems, pp. 23–32 (2012)

55. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Worst-case response-time analysis for mixed messages with offsets in controller area network. In: 17th IEEE Conference on Emerging Technologies Factory Automation, pp. 1–10 (2012)
56. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Extending offset-based response-time analysis for mixed messages in controller area network. In: 18th IEEE Conference on Emerging Technologies Factory Automation, pp. 1–10 (2013)
57. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Extending worst case response-time analysis for mixed messages in controller area network with priority and fifo queues. Access, IEEE **2**, 365–380 (2014)
58. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Response time analysis with offsets for mixed messages in can supporting transmission abort requests. In: Emerging Technology and Factory Automation (ETFA), 2014 IEEE, pp. 1–10 (2014)
59. Mubeen, S., Mäki-Turja, J., Sjödin, M.: Integrating mixed transmission and practical limitations with the worst-case response-time analysis for controller area network. Journal of Systems and Software **99**, 66–84 (2015)
60. Mundhenk, P., Steinhorst, S., Lukasiewycz, M., Fahmy, S.A., Chakraborty, S.: Security analysis of automotive architectures using probabilistic model checking. In: 52nd ACM/IEEE Design Automation Conference (DAC), pp. 1–6 (2015)
61. Natale, M.D.: Evaluating message transmission times in controller area networks without buffer preemption. In: 8th Brazilian Workshop on Real-Time Systems (2006)
62. Navet, N., Song, Y.Q., Simonot, F.: Worst-case deadline failure probability in real-time applications distributed over controller area network. Journal of Systems Architecture **46**(7), 607–617 (2000)
63. Neukirchner, M., Negrean, M., Ernst, R., Bone, T.T.: Response-time analysis of the flexray dynamic segment under consideration of slot-multiplexing. In: 7th IEEE International Symposium on Industrial Embedded Systems, pp. 21–30 (2012)
64. Nolte, T., Hansson, H., Norstrom, C.: Probabilistic worst-case response-time analysis for the controller area network. In: 9th IEEE Real-Time and Embedded Technology and Applications Symposium, pp. 200–207 (2003)
65. Pop, T., Pop, P., Eles, P., Peng, Z., Andrei, A.: Timing analysis of the flexray communication protocol. Real-time systems **39**(1-3), 205–235 (2008)
66. Schenkelaars, T., Vermeulen, B., Goossens, K.: Optimal scheduling of switched flexray networks. In: Design, Automation Test in Europe Conference Exhibition, pp. 1–6 (2011)
67. Schmidt, K., Schmidt, E.: Message scheduling for the flexray protocol: The static segment. IEEE Transactions on Vehicular Technology **58**(5), 2170–2179 (2009)
68. Tanasa, B., Bordoloi, U.D., Kosuch, S., Eles, P., Peng, Z.: Schedulability analysis for the dynamic segment of flexray: A generalization to slot multiplexing. In: 18th IEEE Real-Time and Embedded Technology and Applications Symposium, pp. 185–194 (2012)
69. Tanasa, B., Dutta Bordoloi, U., Eles, P., Peng, Z.: Reliability-aware frame packing for the static segment of flexray. In: Proceedings of the Ninth ACM International Conference on Embedded Software, pp. 175–184 (2011)
70. Thiele, D., Axer, P., Ernst, R.: Improving Formal Timing Analysis of Switched Ethernet by Exploiting FIFO Scheduling. In: Design Automation Conference (DAC). San Francisco, CA, USA (2015)
71. Thiele, D., Diemer, J., Axer, P., Ernst, R., Seyler, J.: Improved formal worst-case timing analysis of weighted round robin scheduling for ethernet. In: In Proc. of CODES+ISSS. Montreal, Canada (2013)
72. Thiele, D., Ernst, R.: Formal Analysis Based Evaluation of Software Defined Networking for Time-Sensitive Ethernet. In: Proc. of Design, Automation, and Test in Europe (DATE). Dresden, Germany (2016)
73. Thiele, D., Ernst, R.: Formal worst-case performance analysis of time-sensitive ethernet with frame preemption. In: Proceedings of Emerging Technologies and Factory Automation (ETFA), p. 9. Berlin, Germany (2016)

74. Thiele, D., Ernst, R.: Formal Worst-Case Timing Analysis of Ethernet TSN's Burst-Limiting Shaper. In: Proc. of Design, Automation, and Test in Europe (DATE). Dresden, Germany (2016)
75. Thiele, D., Ernst, R., Diemer, J.: Formal Worst-Case Timing Analysis of Ethernet TSN's Time-Aware and Peristaltic Shapers. In: IEEE Vehicular Networking Conference (VNC) (2015)
76. Thiele, D., Schlatow, J., Axer, P., Ernst, R.: Formal timing analysis of can-to-ethernet gateway strategies in automotive networks. Real-Time Systems (2015). URL http://dx.doi.org/10.1007/s11241-015-9243-y
77. Tindell, K., Hansson, H., Wellings, A.: Analysing real-time communications: controller area network (CAN). In: IEEE Real-Time Systems Symposium, pp. 259–263 (1994)
78. Vector: CANbedded interaction layer. [Online] http://www.vector.com
79. Yomsi, P., Bertrand, D., Navet, N., Davis, R.: Controller area network (can): Response time analysis with offsets. In: 9th IEEE International Workshop on Factory Communication Systems, pp. 43–52 (2012)
80. Zeng, H., Di Natale, M., Ghosal, A., Sangiovanni-Vincentelli, A.: Schedule optimization of time-triggered systems communicating over the flexray static segment. IEEE Transactions on Industrial Informatics **7**(1), 1–17 (2011)
81. Zeng, H., Di Natale, M., Giusto, P., Sangiovanni-Vincentelli, A.: Stochastic analysis of can-based real-time automotive systems. IEEE Transactions on Industrial Informatics **5**(4), 388–401 (2009)
82. Zeng, H., Di Natale, M., Giusto, P., Sangiovanni-Vincentelli, A.: Using statistical methods to compute the probability distribution of message response time in controller area network. IEEE Transactions on Industrial Informatics **6**(4), 678–691 (2010)
83. Zeng, H., Ghosal, A., Di Natale, M.: Timing analysis and optimization of flexray dynamic segment. In: 7th IEEE International Conference on Embedded Software and Systems, pp. 1932–1939 (2010)

# Index